Railway Ticket Management System

Sree Pramukh Reddy Kamidi (23CSBOB17)

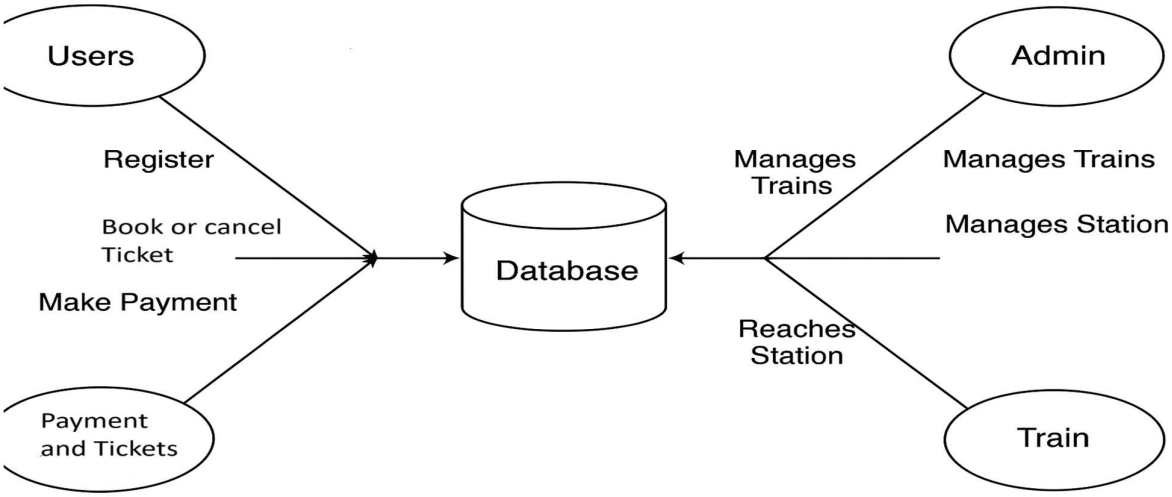
April 26th 2025

Introduction

A **Railway Ticket Management System**, is a software application which helps efficiently organize and manage all the data involved in booking train tickets. From storing passenger details to tracking train schedules and payments. It serves both passengers and railway authorities by digitizing ticket-related operations, reducing manual errors, and improving overall efficiency.

This system allows passengers to easily **search for trains**, **check seat availability**, **book tickets**, **make payments**, and even **cancel bookings** if plans change — all from the comfort of their home. It also keeps users updated with their **ticket status** (like Confirmed, RAC, or Waiting), making travel planning much more predictable and stress-free.

From the railway staff's side, it helps **manage train schedules**, **track passenger information**, **handle payments**, and **monitor station data** efficiently. This reduces paperwork, prevents overbooking, and improves overall service quality.



Problem Statement

We aim to develop a Database System for Online Railway Ticket Registration and Management Platform. The primary responsibilities of this system include:

1. **Allowing users to register and maintain personal profiles** for ticket booking and travel history tracking.
2. **Enabling passengers to book, cancel, and view train tickets** for different trains and travel dates.
3. **Maintaining payment records** of the passengers who have booked the tickets according to the class (2AC/Sleeper etc) details.
4. **Providing admins the ability to manage trains and stations**, update capacities, add or remove routes
5. **Supporting train and route scheduling**, including train numbers, arrival/departure times, and station stops.
6. **Tracking ticket status** (Confirmed, RAC, or Waiting) based on real-time availability and cancellations.

Key Features of the Railway Ticket Management System

Passenger Information Management

The system maintains complete passenger records, including full name, gender, age, contact number, date of birth, and address. Each passenger is uniquely identified and their information can be securely accessed and updated by authorized personnel.

Ticket Booking and Reservation

Passengers can search for trains, check availability, and book tickets. Once booked, each ticket is assigned a unique ID and PNR number, and linked to both the user and the selected train. The system prevents duplicate bookings and ensures seat allocation is managed in real-time.

Train and Schedule Management

Administrators can create and manage train profiles, including train number, name, capacity, and destination. The system also handles route details such as intermediate stations, arrival and departure times, and stop durations, ensuring accurate and conflict-free scheduling.

Payment and Billing System

When a ticket is booked, the system generates a bill and records the amount paid along with the payment method and time. It supports multiple payments per ticket and ensures all payment records are stored securely and can be reviewed when needed.

Ticket Status Tracking

Each ticket is assigned a current status such as Confirmed, RAC, or Waiting, depending on seat availability. The system updates the status automatically based on booking activity and cancellations, allowing passengers to track their position in the queue easily.

Station and Route Mapping

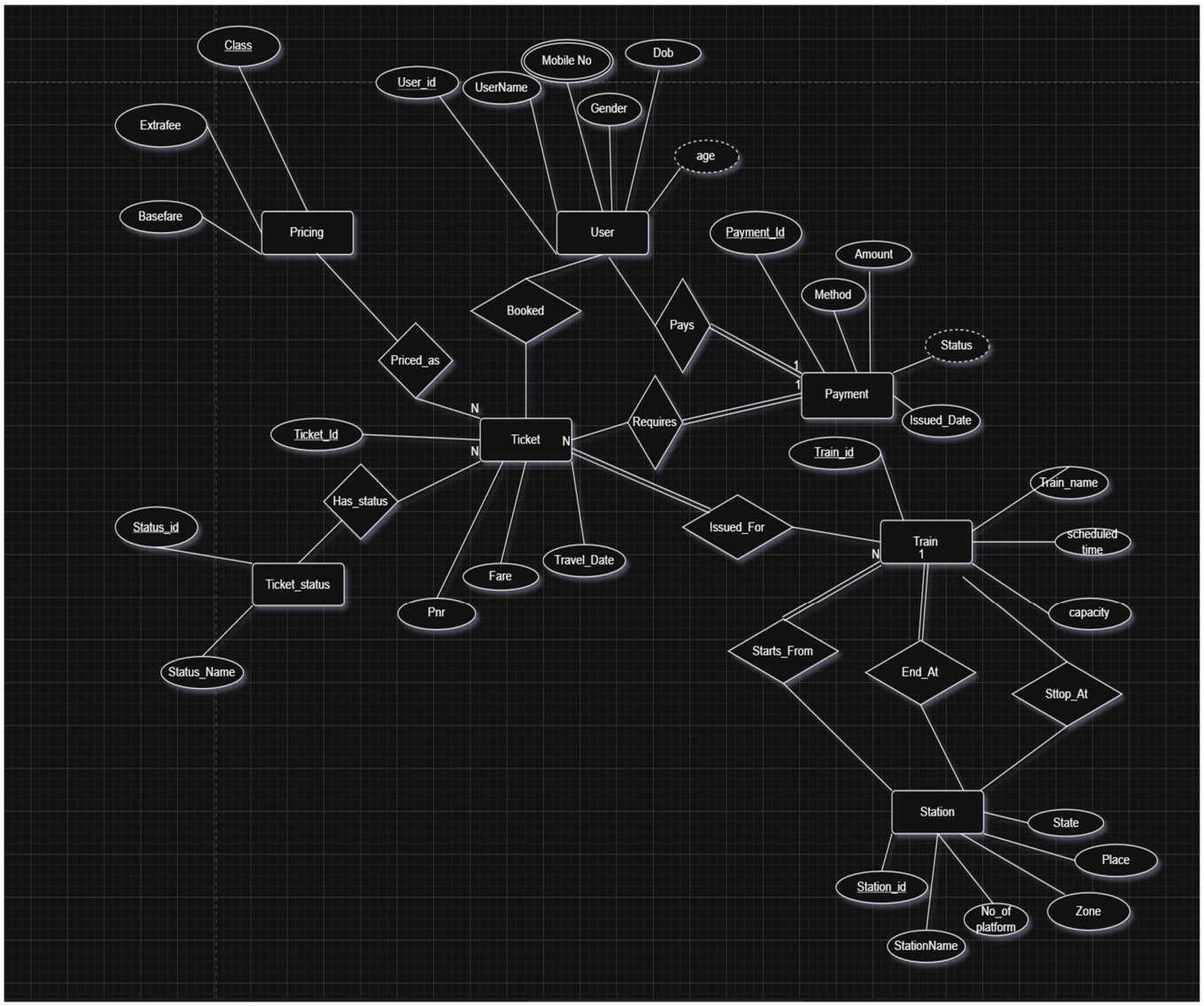
The system maintains a list of all train stations, including station codes, names, and timing details. Each train is linked to one or more stations, and the route is tracked to provide passengers with accurate travel information, including halt times and route coverage.

Admin Control and Monitoring

Administrators have access to manage train data, user records, station schedules, and ticket information. The system offers real-time insights into booking activity, available seats, and

overall system usage, helping in efficient operation and decision-making.

Entity-Relationship Model



ENTITIES

1. USER
    - Represents: a passenger account in the system.
    - Attributes: USER_ID, USERNAME, GENDER, MOBILE_NO, AGE, DOB
    - Primary Key: USER_ID
2. STATION
    - Represents: a railway station where trains start, stop, or end.
    - Attributes: STATION_ID, STATION_NAME, PLACE, STATE, ZONE, NO_OF_PLATFORMS
    - Primary Key: STATION_ID
3. TRAIN
    - Represents: a train service traveling between stations.
    - Attributes: TRAIN_ID, TRAIN_NAME, START_STATION_ID, END_STATION_ID, CAPACITY, SCHEDULED_TIME
    - Primary Key: TRAIN_ID
4. TICKET_STATUS
    - Represents: the booking status of a ticket.
    - Attributes: STATUS_ID, STATUS_NAME
    - Primary Key: STATUS_ID
5. TICKET
    - Represents: a booked journey on a train.

- Attributes: TICKET_ID, PNR, USER_ID, TRAIN_ID, STATUS_ID, TRAVEL_DATE, CLASS, FARE
- Primary Key: TICKET_ID
- Foreign Keys:
  - USER_ID → USER
  - TRAIN_ID → TRAIN
  - STATUS_ID → TICKET_STATUS
  - CLASS → BASE_FARE, EXTRA_FEE

6. PAYMENT
   - Represents: a payment transaction made by a user for a ticket.
   - Attributes: PAYMENT_ID, TICKET_ID, USER_ID, AMOUNT, METHOD, PAYMENT_DATE, PAYMENT_STATUS
   - Primary Key: PAYMENT_ID
   - Foreign Keys:
     - TICKET_ID → TICKET
     - USER_ID → USER
7. PRICING
   - Represents: fare rules per travel CLASS.
   - Attributes: CLASS, BASE_FARE, EXTRA_FARE
   - Primary Key: CLASS

RELATIONSHIPS

1. BOOKED
    - Between: USER (partially participates) and TICKET (totally participates)
    - Cardinality: 1 USER → M TICKET
    - Assumption: A user may book multiple tickets; every ticket must be booked by exactly one user.
    - Implementation: USER_ID (PK of USER) is added as FK in TICKET.
2. STARTS_FROM
    - Between: TRAIN (totally participates) and STATION (partially participates)
    - Cardinality: 1 TRAIN → 1 STATION
    - Assumption: Every train has exactly one start station; a station may be the start for many trains.
    - Implementation: START_STATION_ID (PK of STATION) is added as FK in TRAIN.
3. ENDS_AT
    - Between: TRAIN (totally participates) and STATION (partially participates)
    - Cardinality: 1 TRAIN → 1 STATION
    - Assumption: Every train has exactly one end station; a station may be the end for many trains.
    - Implementation: END_STATION_ID (PK of STATION) is added as FK in TRAIN.
4. STOPS_AT
    - Between: TRAIN (partially participates) and STATION (partially participates)
    - Cardinality: M TRAIN ↔ M STATION

- Assumption: A train may stop at multiple stations; a station may serve many trains.
- Implementation: Associative entity TRAIN_STOP with composite PK (TRAIN_ID, STATION_ID) and attributes ARR_TIME, DEP_TIME, PLATFORM.

5. REQUIRES
    - Between: TICKET (partially participates) and PAYMENT (totally participates)
    - Cardinality: 1 TICKET → M PAYMENT
    - Assumption: A ticket may involve one or more payment transactions; every payment must be linked to one ticket.
    - Implementation: TICKET_ID (PK of TICKET) is added as FK in PAYMENT.
6. PAYS
    - Between: USER (partially participates) and PAYMENT (totally participates)
    - Cardinality: 1 USER → M PAYMENT
    - Assumption: A user may make multiple payments; every payment must be made by one user.
    - Implementation: USER_ID (PK of USER) is added as FK in PAYMENT.
7. HAS_STATUS
    - Between: TICKET (partially participates) and TICKET_STATUS (partially participates)
    - Cardinality: M TICKET → 1 TICKET_STATUS
    - Assumption: Many tickets share the same status; each ticket has exactly one status.
    - Implementation: STATUS_ID (PK of TICKET_STATUS) is added as FK in TICKET.

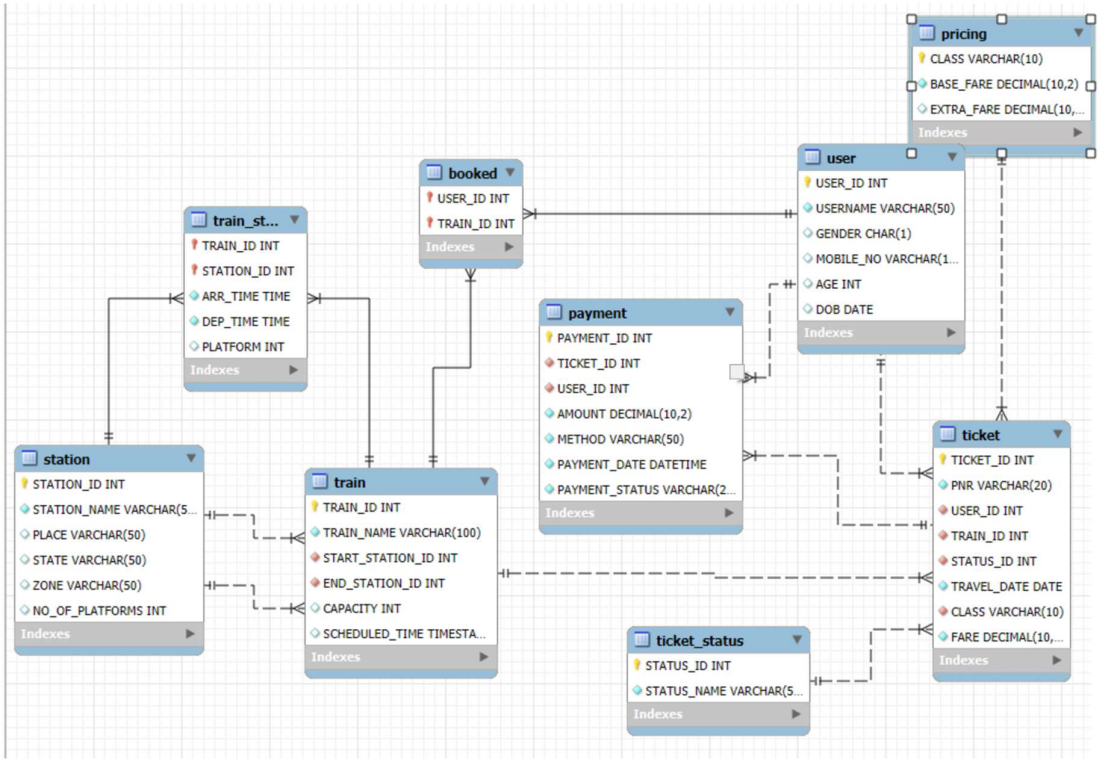
8. ISSUED_FOR

- Between: TICKET (totally participates) and TRAIN (partially participates)
- Cardinality: M TICKET → 1 TRAIN
- Assumption: Every ticket must be booked for exactly one train. A train can have many tickets issued over time.
- Implementation: TRAIN_ID (PK of TRAIN) is added as a foreign key in TICKET.

9. PRICED_AS

- Between: TICKET (partially participates) and PRICING (partially participates)
- Cardinality: M TICKET → 1 PRICING
- Assumption: Many tickets may use the same class (Sleeper, AC, etc.), and pricing is defined per class.
- A ticket refers to a class, and price can be calculated using base + extra fare for that class.
- Implementation: CLASS (PK of PRICING) is added as a foreign key in TICKET.

Relational schema and Normalization



1. USER

FDs:
USER_ID → USERNAME, GENDER, MOBILE_NO, AGE, DOB
Primary Key: USER_ID
Normal Form: Relation is in BCNF

2. STATION

FDs:
STATION_ID → STATION_NAME, PLACE, STATE, ZONE, NO_OF_PLATFORMS
Primary Key: STATION_ID
Normal Form: Relation is in BCNF

3. TRAIN

FDs:
TRAIN_ID → TRAIN_NAME, START_STATION_ID, END_STATION_ID, CAPACITY, SCHEDULED_TIME
Primary Key: TRAIN_ID
Normal Form: Relation is in BCNF

4. TRAIN_STOP

FDs:
(TRAIN_ID, STATION_ID) → ARR_TIME, DEP_TIME, PLATFORM
Primary Key: (TRAIN_ID, STATION_ID)
Normal Form: Relation is in BCNF

5. TICKET_STATUS

FDs:
STATUS_ID → STATUS_NAME
Primary Key: STATUS_ID
Normal Form: Relation is in BCNF

6. TICKET

FDs:
TICKET_ID → PNR, USER_ID, TRAIN_ID, STATUS_ID, TRAVEL_DATE, CLASS, FARE
Primary Key: TICKET_ID
Normal Form: Relation is in BCNF

7. PAYMENT

FDs:
PAYMENT_ID → TICKET_ID, USER_ID, AMOUNT, METHOD, PAYMENT_DATE, PAYMENT_STATUS
Primary Key: PAYMENT_ID
Normal Form: Relation is in BCNF

8. PRICING

FDs:
CLASS → BASE_FARE, EXTRA_FARE
Primary Key: CLASS
Normal Form: Relation is in BCNF

9. TRAIN_CLASS_FARE

(If implemented separately for train-specific pricing)
FDs:
(TRAIN_ID, CLASS) → FARE
Primary Key: (TRAIN_ID, CLASS)
Normal Form: Relation is in BCNF

10. BOOKED

FDs:
(USER_ID, TRAIN_ID) → (no other attributes)
Primary Key: (USER_ID, TRAIN_ID)
Normal Form: Relation is in BCNF

SQL Queries to Create Tables

```
CREATE TABLE USER (
   USER_ID       INT         PRIMARY KEY,
   USERNAME      VARCHAR(50)  NOT NULL,
   GENDER        CHAR(1),
```

```
    MOBILE_NO      VARCHAR(15),
    AGE            INT,
    DOB            DATE
);


CREATE TABLE STATION (
    STATION_ID      INT         PRIMARY KEY,
    STATION_NAME    VARCHAR(50)  NOT NULL,
    PLACE           VARCHAR(50),
    STATE           VARCHAR(50),
    ZONE            VARCHAR(50),
    NO_OF_PLATFORMS INT
);


CREATE TABLE TRAIN (
    TRAIN_ID        INT         PRIMARY KEY,
    TRAIN_NAME      VARCHAR(100) NOT NULL,
    START_STATION_ID INT         NOT NULL,
    END_STATION_ID   INT         NOT NULL,
    CAPACITY        INT,
    SCHEDULED_TIME   TIMESTAMP,
    FOREIGN KEY (START_STATION_ID) REFERENCES STATION(STATION_ID),
    FOREIGN KEY (END_STATION_ID)   REFERENCES STATION(STATION_ID)
);
```

```
CREATE TABLE TRAIN_STOP (
   TRAIN_ID   INT   NOT NULL,
   STATION_ID INT   NOT NULL,
   ARR_TIME   TIME  NOT NULL,
   DEP_TIME   TIME  NOT NULL,
   PLATFORM   INT,
   PRIMARY KEY (TRAIN_ID, STATION_ID),
   FOREIGN KEY (TRAIN_ID)   REFERENCES TRAIN(TRAIN_ID),
   FOREIGN KEY (STATION_ID) REFERENCES STATION(STATION_ID)
);
```

```
CREATE TABLE TICKET_STATUS (
   STATUS_ID   INT         PRIMARY KEY,
   STATUS_NAME VARCHAR(50)  NOT NULL  UNIQUE
);
```

```
CREATE TABLE TICKET (
   TICKET_ID    INT          PRIMARY KEY,
   PNR          VARCHAR(20)   NOT NULL  UNIQUE,
   USER_ID      INT          NOT NULL,
```

```
    TRAIN_ID     INT          NOT NULL,
    STATUS_ID    INT          NOT NULL,
    TRAVEL_DATE  DATE         NOT NULL,
    CLASS        VARCHAR(10)   NOT NULL,
    FARE         DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (USER_ID)   REFERENCES USER(USER_ID),
    FOREIGN KEY (TRAIN_ID)  REFERENCES TRAIN(TRAIN_ID),
    FOREIGN KEY (STATUS_ID) REFERENCES TICKET_STATUS(STATUS_ID),
     FOREIGN KEY (CLASS) REFERENCES PRICING(CLASS)
);

CREATE TABLE PAYMENT (
    PAYMENT_ID     INT          PRIMARY KEY,
    TICKET_ID      INT          NOT NULL,
    USER_ID        INT          NOT NULL,
    AMOUNT         DECIMAL(10,2) NOT NULL,
    METHOD         VARCHAR(50)   NOT NULL,
    PAYMENT_DATE   DATETIME      NOT NULL,
    PAYMENT_STATUS VARCHAR(20)   NOT NULL  CHECK (PAYMENT_STATUS IN
('PAID','PENDING','REFUNDED')),
    FOREIGN KEY (TICKET_ID) REFERENCES TICKET(TICKET_ID),
    FOREIGN KEY (USER_ID)   REFERENCES USER(USER_ID)
);

CREATE TABLE BOOKED (
    USER_ID  INT NOT NULL,
```

```
    TRAIN_ID INT NOT NULL,
    PRIMARY KEY (USER_ID, TRAIN_ID),
    FOREIGN KEY (USER_ID)  REFERENCES USER(USER_ID),
    FOREIGN KEY (TRAIN_ID) REFERENCES TRAIN(TRAIN_ID)
);




CREATE TABLE PRICING (
    CLASS       VARCHAR(10)   PRIMARY KEY,
    BASE_FARE    DECIMAL(10,2) NOT NULL,
    EXTRA_FEE DECIMAL(10,2) default 0.0
);
```

SQL queries to insert data

```
--User
INSERT INTO USER VALUES
(101,'RAMESH','M','99121836899',35,'1990-08-10'),
 (118,'PARDHIV','M','7356282857',20,'2005-06-21'),
(301, 'VENKATESH', 'M', '9845123456', 65, '1960-07-04'),
 (211,'GEETHA','F','9898346582',25,'2000-11-08');
--Station
```

```
INSERT INTO STATION VALUES
(1,'SECUNDERABAD','SECUNDERABAD','TELANGANA','SOUTH',10),

(7,'MUMBAI ','MUMBAI','MAHARASTRA','WEST',8),

(17,'HAZUR SAHIB NANDED','NANDED','MAHARASTRA','WEST',6);




--Train

INSERT INTO TRAIN VALUES

(12733, 'GODAVARI EXPRESS', 1, 7, 1200, '2025-07-05 05:00:00'),

(17299, 'NANDED EXPRESS', 17, 1, 1000, '2025-07-05 09:00:00'),

(14388, 'TIRUPATHI INTERCITY', 21, 8, 800, '2025-07-06 06:30:00');

--Ticket_Status

INSERT INTO TICKET_STATUS VALUES

(1, 'CONFIRMED'),

(2, 'WAITING'),

(3, 'CANCELLED');

--Booked

INSERT INTO BOOKED VALUES

(101, 12733),

(118, 14388),

--Train_Stops
```

```
INSERT INTO TRAIN_STOP VALUES

(12733, 1, '05:00:00', '05:10:00', 1),

(12733, 17, '10:00:00', '10:05:00', 2),

(12733, 7, '20:00:00', '20:10:00', 4);

--Pricing

INSERT INTO PRICING (CLASS, BASE_FARE, EXTRA_FARE) VALUES

('1AC', 2000.00, 300.00),

('2AC', 1500.00, 200.00),

('3AC', 1000.00, 100.00),

('SLEEPER', 500.00, 50.00),

('GENERAL', 200.00, 0.00);



--Ticket
INSERT INTO TICKET VALUES

(1001, 'PNR1001', 101, 12733, 1, '2025-07-06', '1AC', 2300.00),

(1002, 'PNR1002', 118, 14388, 2, '2025-07-07', 'GENERAL', 200.00),

(1003, 'PNR1003', 211, 17299, 1, '2025-07-07', '2AC', 1700.00),



--Payment

INSERT INTO PAYMENT VALUES

(501, 1001, 101, 2300.00, 'CREDIT_CARD', '2025-07-01 14:20:00', 'PAID'),
```

(502, 1002, 118, 200.00, 'UPI', '2025-07-02 09:00:00', 'PENDING'),

(503, 1003, 211, 1700.00, 'DEBIT_CARD', '2025-07-02 10:30:00', 'PAID'),

Database In Action

1. **Passenger Information Management**

i)All Female passengers

```
SELECT *
FROM USER
WHERE GENDER='F';
```

USER_ID	USERNAME	GENDER	MOBILE_NO	AGE	DOB
211	GEETHA	F	9898346582	25	2000-11-08
NULL	NULL	NULL	NULL	NULL	NULL

ii)All senior citizens travelling

```
SELECT *
FROM USER
WHERE AGE>60;
```

USER_ID	USERNAME	GENDER	MOBILE_NO	AGE	DOB
301	VENKATESH	M	9845123456	65	1960-07-04

2. **Ticket Booking and Reservation**

```
SELECT
  U.USERNAME,
  TR.TRAIN_NAME,
  T.CLASS,
  T.TRAVEL_DATE,
  TS.STATUS_NAME
FROM TICKET T
JOIN USER U ON T.USER_ID = U.USER_ID
JOIN TRAIN TR ON T.TRAIN_ID = TR.TRAIN_ID
JOIN TICKET_STATUS TS ON T.STATUS_ID = TS.STATUS_ID
```

ORDER BY T.TRAVEL_DATE, U.USERNAME;

USERNAME	TRAIN_NAME	CLASS	TRAVEL_DATE	STATUS_NAME
RAMESH	GODAVARI EXPRESS	1AC	2025-07-06	CONFIRMED
GEETHA	NANDED EXPRESS	2AC	2025-07-07	CONFIRMED
PARDHIV	TIRUPATHI INTERCITY	GENERAL	2025-07-07	WAITING
GEETHA	GODAVARI EXPRESS	SLEEPER	2025-07-08	CANCELLED
RAMESH	TIRUPATHI INTERCITY	3AC	2025-07-09	CONFIRMED
VENKATESH	GODAVARI EXPRESS	SLEEPER	2025-07-10	CONFIRMED

3.Train and Schedule Management

--Insert new stop

```
INSERT INTO TRAIN_STOP
  (TRAIN_ID, STATION_ID, ARR_TIME, DEP_TIME, PLATFORM)
VALUES
  (17299, 21, '15:00:00', '15:10:00', 2);

-- Verify full route for 17299
SELECT
  S.STATION_NAME,
  TS.ARR_TIME,
  TS.DEP_TIME,
  TS.PLATFORM
FROM TRAIN_STOP TS
JOIN STATION S ON TS.STATION_ID = S.STATION_ID
WHERE TS.TRAIN_ID = 17299
ORDER BY TS.ARR_TIME;
```

STATION_NAME	ARR_TIME	DEP_TIME	PLATFORM
HAZUR SAHIB NANDED	09:00:00	09:10:00	1
GUNTUR	13:00:00	13:05:00	3
TIRUPATHI	15:00:00	15:10:00	2
SECUNDERABAD	17:00:00	17:05:00	5

4.**Payment and Billing System**

```
--list of all tickets booked in July,2025
  SELECT
    T.TICKET_ID,
    T.USER_ID,
    T.FARE,
```

```
   COALESCE(SUM(P.AMOUNT), 0) AS PAID FROM TICKET T
  LEFT JOIN PAYMENT P ON T.TICKET_ID = P.TICKET_ID
  WHERE T.TRAVEL_DATE BETWEEN '2025-07-01'
AND '2025-07-31'
  GROUP BY T.TICKET_ID, T.USER_ID, T.FARE;
```

STATION_NAME	ARR_TIME	DEP_TIME	PLATFORM
HAZUR SAHIB NANDED	09:00:00	09:10:00	1
GUNTUR	13:00:00	13:05:00	3
TIRUPATHI	15:00:00	15:10:00	2
SECUNDERABAD	17:00:00	17:05:00	5

5.Ticket Status Tracking

```
SELECT
  TR.TRAIN_NAME,
  TS.STATUS_NAME,
  COUNT(*) AS NUM_TICKETS
FROM TICKET T
JOIN TRAIN TR ON T.TRAIN_ID = TR.TRAIN_ID
JOIN TICKET_STATUS TS ON T.STATUS_ID = TS.STATUS_ID
WHERE T.TRAVEL_DATE = '2025-07-07'
GROUP BY TR.TRAIN_NAME, TS.STATUS_NAME
ORDER BY TR.TRAIN_NAME, TS.STATUS_NAME;
```

TRAIN_NAME	STATUS_NAME	NUM_TICKETS
NANDED EXPRESS	CONFIRMED	1
TIRUPATHI INTERCITY	WAITING	1